

PERICON a

1. 3 -8bits PORTS A,B & C, each bit being LSTTL compatible, i.e.

input mode: $> 3.3V$ = logical "1" input

$< 0.8V$ = logical "0" input

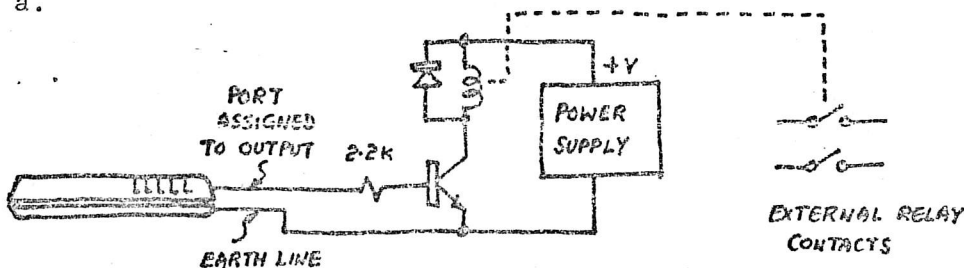
output mode: $> 3.3V$ = logical "1" output

$< 0.8V$ = logical "0" output

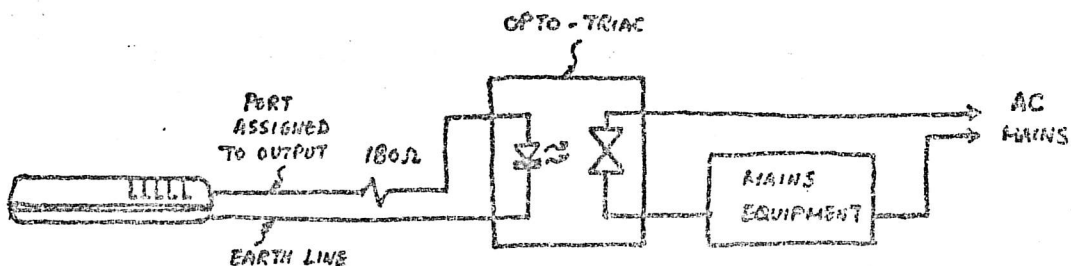
loading being 1 LSTTL equivalent

** note if non-TTL levels can be tolerated, the output can be loaded a lot more. In particular, a logical "1" output on port A or B can source 2 mA into the base input of a Darlington pair driver, or through small limiting resistor (say 180Ω) into a high luminosity LED.

2. An earth line is provided on each of the 2 rows of pin headers. These should be connected to the 0V of any external circuit interacting with PERICON a.



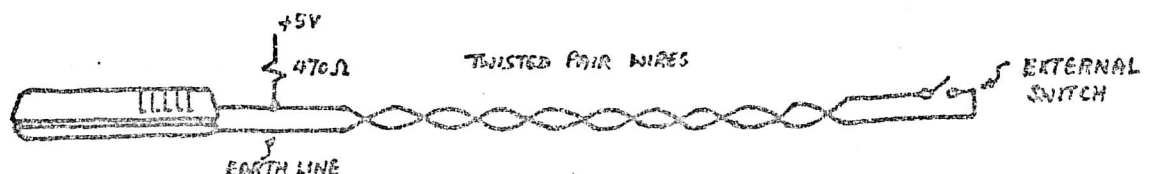
3. To drive mains equipment, it is possible to use a heavy duty relay as suggested in 2 above. A better way would be to use an OPTO-TRIAC, i.e. an optically coupled TRIAC to control mains equipments. This is a much more flexible arrangement and less hazardous. OPTO-TRIAC is available from places like RS Components Ltd. The typical connection would be:



4. There are no pull up resistors on the ports I/O connections. However, all TTL type input will float high, i.e. appears as logical "1" when nothing is connected to it. It is therefore possible to use the circuit below to act as input switch sensor.

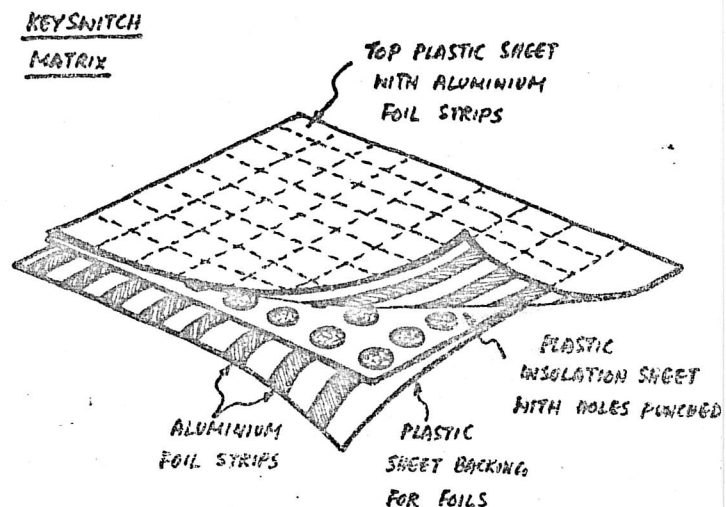
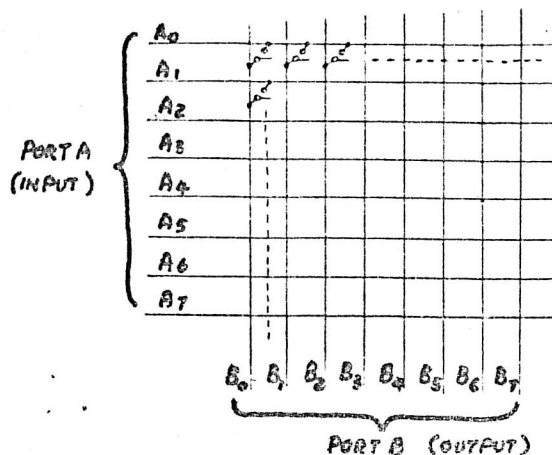


The input impedance is high and would be susceptible to noise triggering if the external switches are more than ~12" away. To operate the input with remote sensors or switches, use the following circuit.



The +5V can be obtained from the rear pin header on the PERSONA module. (Refer to technical sheet on PERSONA about the pin header assignment)

5. One suggested use of PERICON a is to connect it to an external KEYSWITCH matrix, with say PORT A as input and PORT B as output, both using the simple mode 0 format of the 8255-A5 chip.



The KEYSWITCH matrix can be constructed from strips of kitchen aluminium foil stuck down with double sided tape on thin plastic sheet. Two such sheets with 8 strips each are arranged to form the matrix. A single sheet of plastic with holes punched out at each intersection of the top and bottom strips then complete the matrix. Connections to the Aluminium strips can be made with small crocodile clips.

To operate the KEYSWITCHES, a piece of software is needed to write a "0" into PORT B at each of the 8 bits successively, and each time perform a PEEK PORT A to determine if a "0" is detected on PORT A.

(It is necessary to detect "0" as we have mentioned above that an input port will appear as all "1"s if nothing is connected to it. It is then necessary to write "0" on 1 line and "1" on the remaining 7 lines to PORT B in order to provide the scanning action.)

A typical program to replace some functions of the ZX keyboard with a large external keyboard would look like:

```

DIM K$ (8,8)
LET K$ (1,1) = .....
.
.
.
.
LET K$ (8,8) = .....
    } Define the ZX functional code for each of
    } the 64 keys of the KEYSWITCH matrix.

KEYSCAN      POKE PORT B, ( 255 - 2 ** B ) - write A'0' to one column
              LET A = PEEK PORT A           - check if key pressed
              IF A <> 255 THEN GOTO KEYIN     - yes, exit to find which key
              LET B = ( B <= 7 ) * ( B+1 )   - if ( B < 7 ) then B = B+1
                                              else B = 1
              GOTO KEYSKAN                   - loop back to scan next column

KEY IN       LET X = B+1
              LET Y = ( LOG ( 255 - A ) ) / BASE + 1 - get row number from
                                                    value of A.
                                                    Y = ( LOG2 ( 255-A ) ) + 1
              LET KEY = CODE K$ ( X,Y )

```



12 RICKETT STREET
LONDON SW6
TEL: 01-385 2135
TELEX: 8813271 GECOMS G

PERICON a PERICON a PERICON a PERICON a PERICON a PERICON a PERICON a PERICON a PERICON

Below are two Short Subroutines which may be of some use to the PERICON 'a' user. They may be used as they are with perhaps just a few lines of BASIC or they may be incorporated into a Comprehensive Control/Monitoring Program.

The REM Statements may be Junked to save room when necessary.

```
10 rem pericon a INPUT STATUS
15 rem this subroutine will present in k$ an 8 digit representation of the port status.
16 rem port select variable p$ must be set to "a", "b" or "c"
20 if code p$ < 38 or code p$ > 40 then goto 160
30 let p = code p$ - 38
40 let n = 48384
50 poke n + 3, 155
60 let k$ = "00000000"
70 let k = peek (n + p)
80 for i = 7 to 0 step -1
90 if k < (2**i) then goto 120
100 let k$(i+1) = "1"
110 let k = k - 2**i
120 next i
130 print "port "; p$; " bit 01234567"
140 print tab (11); k$
150 return
160 print "port select p$ not set"
170 return
```

```
10 rem pericon a OUTPUT
15 rem this subroutine will convert the 8 digit string n$ to binary and present the data to the port
16 rem k$ must be 8 digits and k$(1) = bit 0 : k$(2) = bit 1 etc.
17 rem port select variable must be "a", "b" or "c"
25 if code p$ < 38 or code p$ > 40 then goto 140
30 let p = code p$ - 38
40 let n = 48384
45 poke n + 3, 128
60 let k = 0
80 if len k$ > 8 then goto 160
90 for i = 8 to 1 step -1
100 let k = (k$(i) = "1") * 2**k
110 next i
120 poke n + p, k
130 return
140 print "port select p$ not set"
150 goto 170
160 print "k$ not 8 digits"
170 return
```